**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ**

**Факультет прикладной математики-процессов управления**

**Программа бакалавриата**

**“Большие данные и распределенная цифровая платформа”**

**ОТЧЕТ**

**по лабораторной работе №3**

**по дисциплине «Системное программирование в Linux»**

**на тему «Разработка системного инструмента для аудита системы»**

**Студент гр. 23Б15-пу**

**Абрахин Е.Д.**

**Преподаватель**

**Киямов Ж. У.**

**Санкт-Петербург**

**2024 г.**

Оглавление

1. [Цель работы 3](#_Toc183725671)
2. [Описание задачи (формализация задачи) 4](#_Toc183725672)
3. [Теоретическая часть 6](#_Toc183725673)
4. [Основные шаги программы 8](#_Toc183725674)
5. [Описание программы 10](#_Toc183725675)
6. [Рекомендации пользователя 11](#_Toc183725676)
7. [Рекомендации программиста 13](#_Toc183725677)
8. [Исходный код программы 13](#_Toc183725678)
9. [Контрольный пример 14](#_Toc183725679)
10. [Вывод 17](#_Toc183725680)
11. [Литература 18](#_Toc183725681)

# Цель работы

Целью данной работы является разработка графического приложения для мониторинга процессов операционной системы. Приложение должно предоставлять пользователю информацию о текущих процессах, их состоянии, а также визуализировать динамику изменения процессов во времени.

# Описание задачи (формализация задачи)

Необходимо создать приложение на Python с использованием библиотек tkinter, matplotlib и psutil, которое будет выполнять мониторинг процессов и предоставлять следующие функции:

1. Мониторинг процессов:

* Определять текущие процессы, запущенные в системе.
* Обновлять информацию о процессах каждые 5 секунд.
* Определять и регистрировать события запуска и завершения процессов.

1. Графический интерфейс:

* Отображать список активных процессов с указанием:
  + Имени процесса.
  + PID (идентификатора процесса).
  + Состояния процесса (работает, спит и т. д.).
* Отдельно выводить лог событий (запуск/завершение процессов).
* Предусмотреть кнопку для ручного обновления списка процессов.

1. Визуализация данных:

* Построение графика изменения количества процессов во времени с разделением по состояниям.
* График должен содержать:
  + Общую линию количества процессов.
  + Линии для каждого состояния процесса (например, running, sleeping и т. д.).
  + Легенду, оси времени и количества процессов.

1. Работа с данными:

* Логировать все события (запуск/завершение процессов) в файл system\_audit.log.
* Сохранять статистику состояний процессов в памяти для построения графиков.

1. Структура приложения:

Использовать два основных вкладки интерфейса:

* Первая вкладка для мониторинга и управления процессами.
* Вторая вкладка для построения и отображения графиков.

1. Используемые библиотеки:

* psutil — для получения информации о процессах.
* tkinter — для создания графического интерфейса.
* matplotlib — для визуализации данных.
* threading — для обеспечения асинхронного мониторинга процессов.

# Теоретическая часть

Программное обеспечение для мониторинга процессов — это инструмент с графическим интерфейсом, позволяющий наблюдать за состоянием процессов в операционной системе в режиме реального времени. Приложение регистрирует события запуска и завершения процессов, предоставляет статистику об их динамике и визуализирует данные в удобной форме. Основной целью является создание эффективного и удобного инструмента для анализа и мониторинга процессов.

Основы работы с системами мониторинга процессов

Системы мониторинга процессов позволяют получать информацию о запущенных приложениях, их текущем состоянии и других параметрах (например, PID, использование ресурсов). Такое программное обеспечение полезно для анализа производительности системы, диагностики и управления задачами.

В рамках рассматриваемого приложения выполняются следующие функции:

* Регистрация текущих процессов, их статуса и идентификатора.
* Обнаружение изменений в списке процессов (новые или завершённые).
* Логирование событий (запуск/завершение процессов).
* Построение графиков изменения статистики по процессам во времени.

Алгоритмы работы с процессами

Для мониторинга процессов используется библиотека psutil, обеспечивающая доступ к данным о процессах. Основные этапы работы:

1. Сбор данных:

* Каждые 5 секунд приложение получает информацию о всех активных процессах.
* Информация включает имя процесса, его PID и статус (например, running, sleeping).

1. Определение изменений:

* Сравниваются текущий список процессов с предыдущим:
* Добавляются новые процессы.
* Удаляются завершённые процессы.
* Эти события логируются в файл system\_audit.log.

1. Подсчёт статистики:

* На каждом шаге подсчитывается количество процессов в различных статусах.
* Данные сохраняются для последующего анализа и построения графиков.

Архитектура приложения

Приложение состоит из трёх ключевых компонентов:

1. Графический интерфейс пользователя (GUI):

Реализован с использованием библиотеки tkinter.

Вкладка для мониторинга включает:

* Список активных процессов с указанием имени, PID и состояния.
* Лог событий запуска и завершения процессов.

Вкладка для отчётов включает:

* Кнопку генерации графика статистики процессов.
* Поле для отображения визуализации (графики).

1. Модуль мониторинга:

Реализован как отдельный поток (с помощью threading) для постоянного обновления информации.

Обрабатывает исключения, связанные с доступом к процессам.

1. Визуализация данных:

Для построения графиков используется библиотека matplotlib.

График отображает:

* Общую динамику количества процессов.
* Количество процессов в каждом состоянии (running, sleeping и т. д.).
* В интерфейсе предусмотрена возможность визуального анализа этих данных.

1. Результаты:

* Вкладка с графиком позволяет анализировать изменения в количестве процессов и их состояниях.
* Логи сохраняются в файл для дальнейшего использования.

# Основные шаги программы

1. Запуск мониторинга процессов

При запуске программы активируется фоновый поток для отслеживания всех текущих процессов.

Используя библиотеку psutil, программа собирает информацию о каждом процессе, включая:

* Идентификатор (PID).
* Имя процесса.
* Текущий статус (например, running, sleeping).

2. Логирование событий процессов

Для новых процессов фиксируются события запуска, а для завершённых — события завершения.

Все события записываются в файл system\_audit.log с указанием времени.

Логирование также отображается в реальном времени в интерфейсе (список событий).

3. Графический интерфейс

Реализован с использованием библиотеки tkinter и состоит из двух вкладок:

Вкладка "Process Monitoring" (Мониторинг процессов) отображает текущий список активных процессов в формате:

* Имя процесса, PID, статус.

Лог событий отображается в отдельном окне.

Кнопка "Update Process List" обновляет список процессов вручную.

Вкладка "Generate Report" (Создание отчёта):

Включает элементы управления для визуализации данных.

Кнопка "Show Report" строит графики изменения количества процессов и их состояний.

4. Формирование статистики

Программа собирает информацию каждые 5 секунд:

Общее количество процессов.

Распределение процессов по состояниям (например, running, sleeping).

Эти данные сохраняются для построения графиков.

5. Построение отчёта

С помощью библиотеки matplotlib создаётся график, отображающий:

Общую динамику числа процессов во времени.

Динамику изменения процессов в различных состояниях.

График обновляется при каждом вызове функции отображения отчёта.

6. Взаимодействие с пользователем

Пользователь может:

Наблюдать за текущим состоянием процессов.

Генерировать графики с отчётами.

Сохранять логи событий для анализа.

7. Завершение работы программы

Программа завершает мониторинг автоматически при закрытии.

Все данные логирования остаются доступными в файле system\_audit.log.

# Описание программы

Программа представляет собой графическое приложение для мониторинга системных процессов и визуализации их статистики. Разработана на Python 3.12.6 с использованием библиотек psutil, tkinter, matplotlib и logging. Основное назначение программы — предоставлять пользователю информацию о текущих системных процессах, фиксировать изменения и отображать статистику в графическом виде.

Таблица 1. AuditTool.py

|  |  |  |  |
| --- | --- | --- | --- |
| Функция | Описание | Аргументы | Возвращаемое значение |
| monitorSystem | Отслеживает системные процессы, фиксирует изменения и записывает события в лог. | Без аргументов | None |
| logMessage | Записывает событие в лог-файл и выводит сообщение в список событий интерфейса. | message (строка) | None |
| updateProcessList | Обновляет список текущих процессов в интерфейсе. | Без аргументов | None |
| showReport | Генерирует и отображает график изменения статистики процессов во времени. | Без аргументов | None |

# Рекомендации пользователя

Программа представляет собой графический интерфейс для мониторинга и анализа системных процессов. Ниже приведены инструкции по работе с программой:

1. Запуск программы

Перед началом работы убедитесь, что у вас установлены все необходимые зависимости (psutil, matplotlib, tkinter). Программа автоматически начнет мониторинг системных процессов после запуска.

1. Мониторинг процессов

После запуска программы откройте вкладку "Process Monitoring":

Список процессов:

* + Отображается перечень текущих активных процессов с их PID и статусами.
  + Для обновления списка вручную нажмите кнопку "Update Process List".

События процессов:

* + В правой части окна выводятся логи событий:
    - Появление новых процессов.
    - Завершение текущих процессов.
  + Логи также сохраняются в файл system\_audit.log в каталоге программы.

1. Генерация отчета

Перейдите на вкладку "Generate Report", чтобы получить визуализацию статистики процессов:

Нажмите кнопку "Show Report" для построения графика.

На графике отображается:

* Общее количество процессов.
* Количество процессов в различных состояниях (например, running, sleeping).

Подробная информация:

Рядом с графиком отображаются численные данные о процессах на текущий момент времени.

1. Особенности работы

Программа автоматически обновляет статистику каждые 5 секунд.

Статистика включает:

* Количество всех процессов.
* Разбиение по статусам (running, sleeping, и т.д.).
* Временные метки.

Обработка ошибок:

Если программа не может получить доступ к процессу, она продолжает работу, игнорируя недоступный процесс.

5. Логирование событий

Все события, включая запуск и завершение процессов, записываются в лог-файл:

Формат записи: [Дата и время] Сообщение.

Лог хранится в файле system\_audit.log в каталоге программы.

6. Выход из программы

Просто закройте окно приложения, чтобы остановить мониторинг. Все сохраненные данные останутся доступны для анализа

# Рекомендации программиста

Для успешного запуска программы убедитесь, что у вас установлен Python версии не ниже 3.12.0 и необходимые библиотеки, такие как psutil, matplotlib и tkinter.

Поддерживайте актуальные версии используемых библиотек и Python для обеспечения работоспособности программы на современных системах. Следуйте передовым практикам разработки, уделяя внимание четкому именованию переменных и функций, чтобы улучшить читаемость и понимание кода.

Регулярно проводите тестирование программы на различных входных данных, чтобы убедиться в ее надежности и корректности работы. Рассмотрите возможность добавления логирования для отслеживания процесса выполнения и отладки. Используйте комментарии для пояснения сложных участков кода и логики программы.

# Исходный код программы

[https://github.com/FasterXaos/System\_programming\_in\_Linux.git](https://github.com/FasterXaos/System_programming_in_Linux.git%20)

# Контрольный пример

1. Запуск программы

Запустите файл AuditTool.py.

Откроется окно с двумя вкладками (Рис. 1):

Process Monitoring – для просмотра текущих процессов.

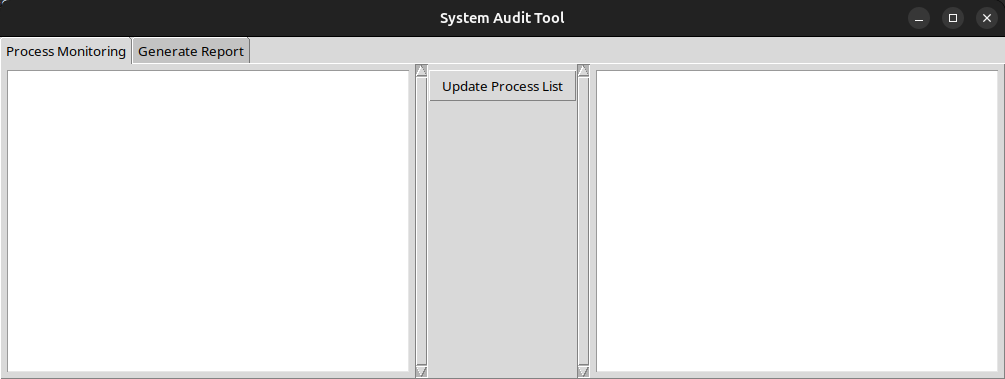
Generate Report – для построения отчета. 

Рис. 1. Пример стартового окна программы

1. Мониторинг процессов

На вкладке Process Monitoring:

Слева отображается список текущих процессов (название, PID, статус).

Справа – журнал событий: запуск или завершение процессов.

Для обновления данных нажмите кнопку "Update Process List" (Рис. 2).

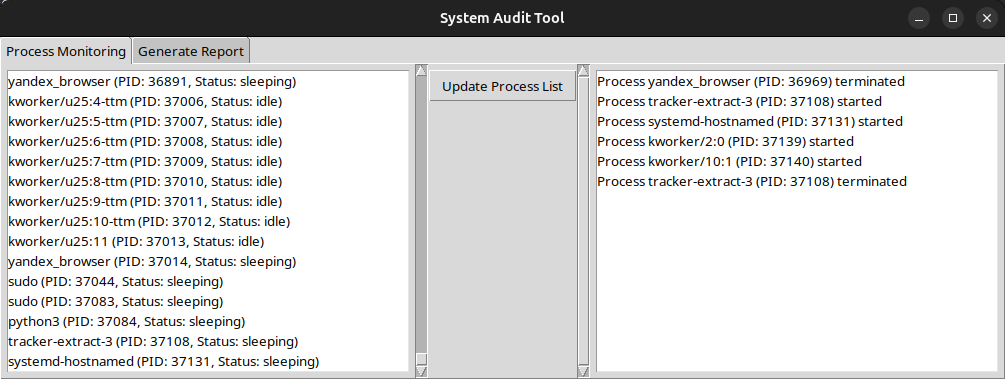


Рис. 2. Пример работы в окне Process Monitoring

1. Генерация отчета

Перейдите на вкладку Generate Report:

Нажмите "Show Report" для построения графика.

На графике отобразится:

Общее количество процессов.

Количество процессов в разных статусах.

Рядом с графиком выводится информация о текущем состоянии процессов (Рис. 3).

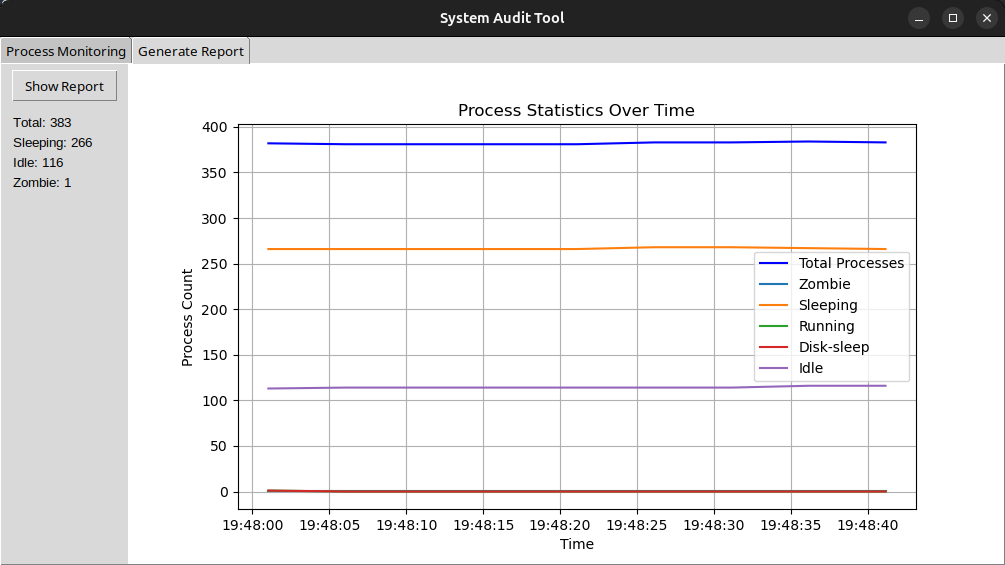


Рис. 3. Пример работы в окне Generate Report

1. Логирование событий

Все изменения фиксируются в файле system\_audit.log.

Формат: [Дата и время] Сообщение.

# Вывод

В процессе работы была разработана система мониторинга процессов, которая позволяет:

* Отслеживать активные процессы на устройстве (название, PID, статус).
* Фиксировать изменения (запуск и завершение процессов) в реальном времени.
* Генерировать графические отчеты, отображающие динамику изменения количества процессов и их состояний (например, активный, приостановленный).
* Логировать события в файл system\_audit.log для последующего анализа.

Программа разработана на Python с использованием следующих библиотек:

* tkinter – для реализации графического интерфейса.
* psutil – для получения данных о процессах.
* matplotlib – для построения графиков.

Система предоставляет удобный интерфейс для взаимодействия с пользователем, включая просмотр процессов, регистрацию событий и визуализацию статистики.

# Литература

1. psutil – Process and System Utilities // GitHub Documentation

URL: https://github.com/giampaolo/psutil (дата обращения: 1.12.2024).

1. matplotlib – Visualization with Python // Matplotlib Documentation

URL: https://matplotlib.org/stable/index.html (дата обращения: 1.12.2024).

1. tkinter – Python interface to Tcl/Tk // Python Documentation

URL: https://docs.python.org/3/library/tkinter.html (дата обращения: 1.12.2024).